Project 1

Huzaifa Jahan (215668106)
huzaifaj@my.yorku.ca

April 26, 2021

Note: This is the project for 4404 students only. You need to work individually for this project. You must use this latex template to write up your report. Remember to fill in your information (name, student number, email) at above. Submit your codes/scripts (*.zip) and a project report (*.pdf) (maximum 6 pages) from eClass before the deadline. No late submission will be accepted. No handwriting is accepted. Direct your queries to Hui Jiang (hj@eecs.yorku.ca).

Latent Semantic Analysis for Natural Language Processing

In this project, you will use a text corpus, called the English Wikipedia Dump, to construct document-word matrices and then use the latent semantic analysis (LSA) technique to factorize the matrices to derive word representations, a.k.a. *word embeddings* or *word vectors*. You will first use the derived word vectors to investigate semantic similarity between different words based on the Pearson's correlation coefficient obtained by comparing cosine distance between word vectors and human assigned similarity scores in a data set called *WordSim353* (http://www.cse.yorku.ca/~hj/wordsim353_human_scores.txt). Furthermore, the derived word vectors will be visualized in a 2-dimensional space using the t-SNE method to inspect the semantic relationship among English words. In this project, you will implement several machine learning methods to factorize large sparse matrices to study how to produce meaningful word representations for natural language processing.

1. Use a small data set, called *enwiki8* (downloading from http://www.cse.yorku.ca/~hj/enwiki8.txt.zip) to construct a document-word frequency matrix in Figure 7.7. In this project, you treat each paragraph in a line as a document. You construct the matrix in a sparse format for the top 10,000 most frequent words in *enwiki8* and all words in *WordSim353*.

2. You first use a standard SVD procedure from a linear algebra library to factorize the sparse document-word matrix, and truncate it to $k = 20, 50, 100$. Examine the run-in time and memory consumption for SVD.

3. You can choose any algorithm from the following two choices to implement matrix factorization from scratch[1]:

   (a) The alternating Algorithm 7.6; or
   (b) The stochastic gradient descent (SGD) method in Exercise 7.5.

   Use your implementation to factorize the document-word matrix for $k = 20, 50, 100$. Examine the run-in time and memory consumption.

4. Investigate the quality of the above derived word vectors based on the correlation with some human assigned similarity scores. For each pair of words in *WordSim353*, compute the cosine distance between their word vectors and then compute the Pearson's correlation coefficient between these cosine distances and human scores. Tuning your learning hyperparameters towards higher correlation.

[1] You are only allowed to use linear algebra libraries, such as matrix multiplication, matrix inversion, eigenvalues and eigenvectors.

5. Visualize the above word representations for the top 300 most frequent words in *enwiki8* using the t-SNE method[2] by projecting each set into a 2-dimensional space. Investigate how these 300 word representations are distributed and inspect whether the semantically relevant words are located closer in the space. Explain why.

6. Refer to [1] to re-construct the document-word matrix based on the positive pointwise mutual information (PPMI). Repeat the above steps 2-5 to see how much the performance is improved.

What to submit?

You need to submit all of your codes written for this project. Please provide a clear instruction on how to repeat your experiments in a separate readme file. You need to submit a project report (in pdf, maximum 6 pages) to summarize what you have done in terms of algorithm development and experimental fine-tuning, also report the best settings for each case and discuss your findings from this project.

References

[1] Peter D. Turney and Patrick Pantel. 'From Frequency to Meaning: Vector Space Models of Semantics.' In: *J. Artif. Int. Res.* 37.1 (Jan. 2010), pp. 141–188. (`https://arxiv.org/abs/1003.1141`)

[2] You can use any existing t-SNE library.

Your Report (maximum 6 pages):
(LSA) is a technique in NLP, in particular distributional semantics, of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. You could call it "Topic Modeling". In the project we used different methods to filter the corpus and further factorize our sparse matrix and finally finding higher correlation in topics using Pearson coefficient. For a sparse matrix is very beneficial as it allows us to bypass partial memory issues with normal matrix and more function processing. A majority of the project felt as though we were attempting to make the run time of the program more efficient with respect to topic extraction. When testing various methods to form the matrix storing became an issue and the usage of stop words and filtering became key. SVD usage with 20 50 100 affect ted run time very minimally and memory consumption was fine at 50. The use of the SGD algorithm allows us to factorize the matrix and reduce dimensionality to allow for easier processing in run time/memory consumption. Positive point wise mutual information helps in the Pearson coo efficient to help make it close to 50 and more precise but provides a small but noticeable difference in run time over all.